

## **TEMA 1: “Entorn de programació i primers programes.”**

### **BASE TEÒRICA**

- Instal·lació del java (SDK) i l'IDE Netbeans

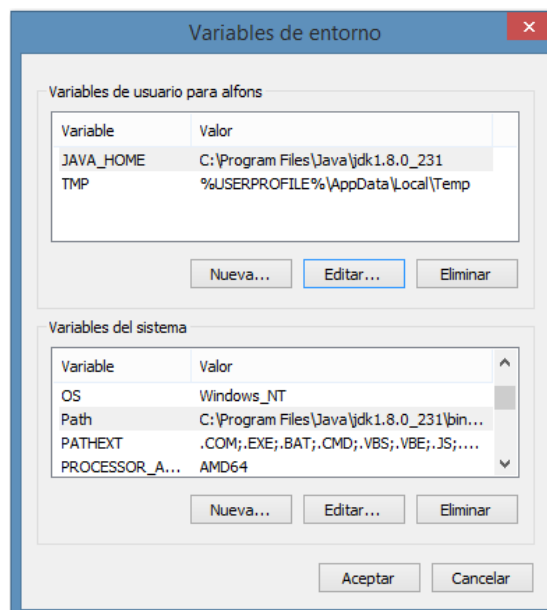
<https://www.oracle.com/java/technologies/javase/javase8u211-later-archive-downloads.html>

<https://www.oracle.com/java/technologies/downloads/archive/#docs>

[https://filehippo.com/es/download\\_netbeans/8.2/](https://filehippo.com/es/download_netbeans/8.2/)

A l'hora de instal·lar el paquet java SDK es recomana configurar les variables PATH del sistema operatiu:

- afegint el directori bin, al principi de la llista, acabant en punt i coma.
- afegint variables del sistema JAVA\_HOME (indica la carpeta principal de java SDK)



- Estructura bàsica d'un programa en Java:

```
// import javax.swing.JOptionPane;  
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
        //JOptionPane.showMessageDialog(null, "Hello World");  
    }  
}
```

- Edició i compilació d'un programa en java SDK.

1.- Editar el programa: crear una classe Java en un arxiu .java : el codi de la classe, ha d'estar dins de l'arxiu amb el mateix nom de la classe i extensió Java: "HelloWorld.java". S'ha d'usar notepad o qualsevol altre editor de text pla.

2.- Compilar el programa en java: per que la classe sigui entesa per la màquina virtual de java (JVM), és necessari compilar-la. Per això usarem el compilador javac.exe, que ve dins del paquet SDK.

```
C:\>javac HelloWorld.java
```

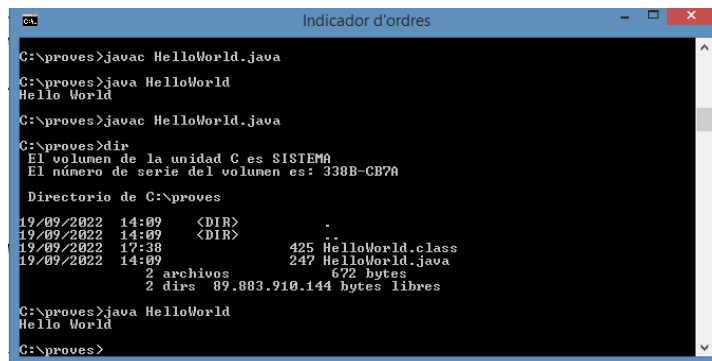
El compilador, agafarà l'arxiu HelloWorld.java i generarà un arxiu HelloWorld.class, que conte el bytecode, que és el que entenc la màquina virtual java.

3.- Executar el programa: mitjançant la màquina virtual de Java: la JVM executarà l'arxiu HelloWorld.class a partir del nom de la classe:

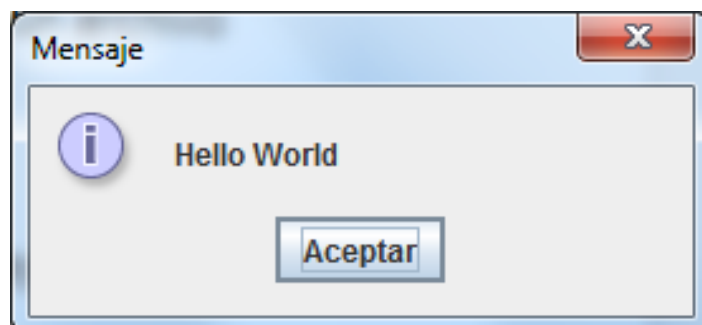
```
C:\>java HelloWorld
```

Cal destacar que Java, buscarà la classe en el directori actual, per la qual cosa s'ha d'executar en el mateix directori on es troba l'arxiu .class.

Veurem un missatge "Hello World" o una finestra que diu "Hello World", segons el codi compilat:

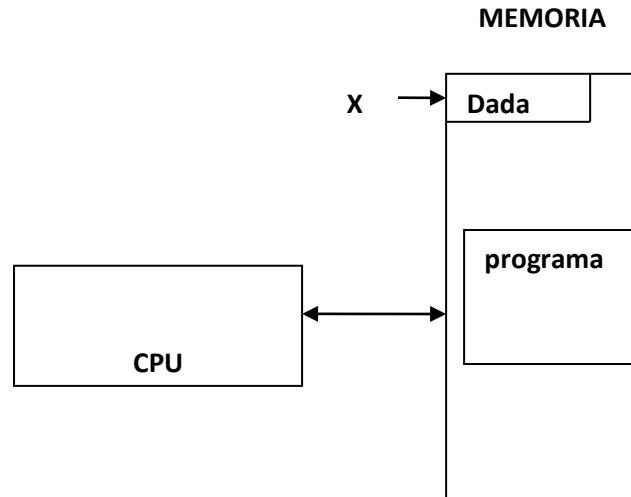


```
C:\proves>javac HelloWorld.java
C:\proves>java HelloWorld
Hello World
C:\proves>javac HelloWorld.java
C:\proves>dir
El volumen de la unidad C es SISTEMA
El número de serie del volumen es: 338B-CB7A
Directorio de C:\proves
19/09/2022  14:09    <DIR>          .
19/09/2022  14:09    <DIR>          ..
19/09/2022  17:38                425 HelloWorld.class
19/09/2022  14:09                247 HelloWorld.java
               2 archivos            672 bytes
               2 dirs  89.883.910.144 bytes libres
C:\proves>java HelloWorld
Hello World
C:\proves>
```



- Concepte de variable. Declaració, inicialització i tipus.

Una variable és un espai de memòria referenciat per l'ordinador:



Declaració: *<tipus>* *<identificador>*

*<tipus>* : **int** (nombre enter)  
**double** (nombre real)  
**char** (caràcter alfanumèric)  
**boolean** (booleà)  
**String** (cadena de caràcters)

Exemple: **int** edat = 10;

Les dades numèriques tenen un rang i per tant són finits.

Tipus	Descripció	Mida	Rang
int	Nombres enters	4 bytes	-2,147,483,648 a 2,147,483,647
double	Nombres reals	8 bytes	$4.9 \times 10^{-324}$ a $1.7976931348623157 \times 10^{308}$
boolean	Booleans	1 bit	true / false
char	Caràcters simples	2 bytes	Unicode UTF-16 \u0000 a \uffff

Una constant és una variable de valor fix al llarg de tot el programa, es declaren com a static (de classe) i final (immodificables), assignant el valor a la declaració.:

**static final int** EDAT = 10; //el nom es posa en majúscules normalment

Les variables del tipus **int** ocupen 4 octets (bytes) . Permeten l'ús d'operadors (short i long ) i es representen internament en binari.

Les variables tipus **char** contenen un únic caràcter i s'emmagatzemen en dos bytes (16 bits). Per tant, una variable char pot contenir  $2^{16} = 65.536$  valors diferents: en decimal de 0 a 65.535, en hexadecimal: 0000 a FFFF. La correspondència es fa a través del codi Unicode UTF-8::

<https://unicode-table.com/es/>

Cal destacar que el codi ASCII esta contingut en Unicode, però aquest últim proporciona molts més caràcters (diferents idiomes) ja que el codi ASCII com a màxim conté  $2^8=256$  caràcters. Podem utilitzar els caràcters literals 'a', '\$' o 'b', però també podem referenciar-los directament pel seu codi Unicode en format hexadecimal: \u0061, \u0024 i \u0062 respectivament. Noteu que el codi Unicode que representa un caràcter existent en el codi ASCII es el mateix precedit d'un byte en 0, es a dir 'a' és el número 61h del codi ASCII i és el número 0061h del codi Unicode.

Per últim s'ha de tenir en compte en base a la codificació comentada que una variable del tipus char es pot considerar com un enter i, per tant, estan permeses les mateixes operacions que entre els enters.

Les variables **double** (també anomenades de punt flotant) permeten representar nombres amb una part entera i una part fraccionària. Aquestes variables es representen internament per mitjà de la mantissa, i l'exponent, que representa la potència de 10 que multiplica a la mantissa. El nombre quedarà definit com: mantissa per 10 elevat a l'exponent

Una variable de tipus **bool** només pot tenir els valors false o true. Aquestes variables son una alternativa pel 0 ( zero ) i indicar fals i diferent de 0 ( zero ) per indicar veritat.

Un **string** representa una seqüència de caràcters: "Hola mon".

- Entrada i sortida de dades en Java:

Escriure dades en pantalla:

```
System.out.print(<expressió>);  
System.out.println(<expressió>);
```

Introduir dades des del teclat: Es declara un objecte Scanner tipus System.in.

```
Scanner <variable> = new Scanner(System.in);
```

Per llegir dades s'ha de fer servir una variable i utilitzar el mètode compatible amb el seu tipus:

```
String nom = <variable>.nextLine();  
int edad = <variable>.nextInt();  
double x = <variable>.nextDouble();
```

S'aconsella llegir les dades com a String i convertir-les a integer o double quan sigui necessari:

```
String num = <variable>.nextLine();  
int edad = Integer.parseInt(<variable>);  
double x = Double.parseDouble(<variable>);
```

- Llibreries (API de Java)

El SDK de Java inclou la màquina virtual de Java (JRE) i el compilador, però també inclou entre d'altres utilitats el que es coneix com API – "**Application Programming Interface**", que és un conjunt de llibrerías de codi Java compilat o classes ja preparades per que siguin usades pels programadors. Aquestes llibreries es classifiquen en paquets (*package*) de classes i s'inclouen als programes amb la instrucció: **import**

Per exemple, per poder utilitzar la classe Scanner haurem de incloure als nostres programes la classe **Scanner** dins del paquet **util** :

```
import.util.Scanner; equivalent a : import.util.*;
```

on l'asterisc inclou totes les classes dins d'un paquet (conjunt de classes).

La classe **Math** proporciona dos constants (**Math.E** i **Math.PI**) y molts mètodes (de classe) útils per a operacions i funcions matemàtiques complexes. És una classe del llenguatge (java.lang) i per això s'importa automàticament (per defecte a tots els programes).

Alguns dels mètodes de la classe Math són:

<i>Math.abs(num)</i>	<i>Math.acos(num)</i>	<i>Math.atan(num)</i>
<i>Math.cos(num)</i>	<i>Math.exp(num)</i>	<i>Math.log(num)</i>
<i>Math.max(a,b)</i>	<i>Math.min(a,b)</i>	<i>Math.pow(a,b)</i>
<i>Math.random()</i>	<i>Math.round(num)</i>	<i>Math.sin(num)</i>
<i>Math.sqrt(num)</i>	<i>Math.tan(num)</i>	<i>Math.toDegrees(num)</i>
<i>Math.toRadians(num)</i>		

Els paquets estàndard de l'API de Java són :

java.lang	// classes i interfases bàsiques (s'importa per defecte)
java.applet	// classe Applet i interfases per a interacció amb el navegador
java.awt	// Abstract Windowing Toolkit: classes gràfiques antigues
java.io	// Entrada/Sortida
java.net	// classes per a comunicació a través de protocols Internet
java.rmi	// Programació distribuïda
java.security	// Seguretat en Java
java.util	// Utilitats diverses
Javax.swing	// components gràfics moderns per a Java

La API de Java esta documentada en línia des de:

<http://DirectoriInstalacioJava/docs/api/index.html>

i també des de l'adreça:

<https://docs.oracle.com/javase/8/docs/api/index.html?overview-summary.html>

- Operadors i expressions.

Els operadors especifiquen una avaluació que es realitzarà en un o més operands, es poden classificar en:

<u>Aritmètics:</u>	<b>Suma:</b> +
	<b>Resta:</b> −
	<b>Multiplicació:</b> *
	<b>Divisió:</b> /
	<b>Mòdul (residu):</b> %
<u>Assignació:</u>	<b>operador d'assignació:</b> =

Aquest operador atribueix a una variable –es a dir, diposita a la zona de memòria corresponent – el resultat d'una expressió o el valor d'una altra variable. No s'ha de confondre amb la igualtat matemàtica. La seva forma general és:

**<nom\_de\_la\_variable> = <expressió>;**

I el que fa és: s'avalua *expressió* de la part esquerra i el resultat es posa en *nom\_de\_la\_variable de la part dreta*, substituint qualsevol altre valor que hi hagués en aquesta posició de memòria assenyalada per la variable.

Sovint es fa:  $x = x + 1$ ; que equival a l'expressió compacta:  $x++$ ;

**Relacionals:** Igual que: =  
Menor que: <  
Major que: >  
Menor o igual que: <=  
Major o igual que: >=  
Diferent de: !=

**Lògics:** Operador Y : && o and  
Operador OR: || o or  
Operador NOT: !

**Altres:** Increment i decrement : ++, --

*Exemple:* `i = 2;`  
`j = 2;`  
`m = i++;` // m=2 , i=3  
`n = ++j;` // n=3 , j=3

Operador punt (.)  
Operador instància de classe ; instanceof

Exemples:

Operator	Function	Example	Result
+	Add	<code>int i = 10 + 2;</code>	12
-	Subtract	<code>int j = i - 3;</code>	9
/	Divide	<code>double k = j / 3;</code>	3.00
*	Multiply	<code>int product = i * j;</code>	108
++	Add 1	<code>i++;</code>	13
--	Subtract 1	<code>j--;</code>	8
%	Modulus	<code>int m = 12 % 5;</code>	2

- Expressions

Una **expressió** és una combinació de variables i/o constants, i operadors. L'expressió és equivalent al resultat que proporciona al aplicar els seus operadors als seus operands. Hi ha diversos tipus d'expressions:

*Aritmètiques:* Segueixen les normes aritmètiques convencionals  
*Lògiques:* Equivalen sempre a un valor 1 (true) o 0 (false)  
*Generals:* Qualsevol combinació de les dos anteriors

El resultat d'una expressió lògica és un valor numèric (1 o 0) això permet que qualsevol expressió lògica pugui aparèixer com a sub-expressió en una expressió aritmètica. Igualment, qualsevol valor numèric pot ser considerat com un valor lògic: **true** si és diferent de 0 i **false** si és igual a 0, la qual cosa permet introduir qualsevol expressió aritmètica com a sub-expressió d'una expressió lògica. Per exemple:

`(a - b*2.0) && (c != d)`

Els operadors tindran una jerarquia a l'hora de ser avaluats sempre i quan no hi hagi parèntesi a les expressions:



Mètodes de la classe String més utilitzats :

Method	Function	Example
<code>.charAt (x)</code>	returns the char from a specified index	<code>String colour = "blue"; char letter = colour.charAt (0) ;</code>
<code>.toUpperCase ()</code>	returns the String in UPPER CASE	<code>String name = "bob"; bob = bob.toUpperCase () ;</code>
<code>.toLowerCase ()</code>	returns the String in lower case	<code>String pet = "DOG"; pet = pet.toLowerCase () ;</code>
<code>.substring (x,y)</code>	returns String portion between two indexes	<code>String s = "I love hats"; String snip = s.substring (2,6) ;</code>
<code>.length ()</code>	returns how many characters there are in a String	<code>String h = "radar"; int size = h.length () ;</code>

Realment String és una classe de java i no un tipus de dada predefinit, per la qual cosa té els mètodes anteriors i d'altres. Precisament degut a que és una classe a l'hora de fer comparacions entre cadenes de caràcters ho hem de fer mitjançant el mètode **.equals**:

**cadena1.equals(cadena2);**

que ens tornarà true si son iguals i false si no ho son.

- Comentaris i javadoc

Hi ha tres tipus de comentaris:

// D'una sola línia

/\* De una ó  
més línies  
\*/

/\*\* De documentació, d'una o diverses línies.  
Just abans de l'element que documenta (classe, mètode, etc )  
\*/

Els comentaris de documentació son utilitzats per l'eina **javadoc** per a generar la documentació del programa en format HTML a partir de certes etiquetes:

Per a referències: **@see** <altra classe>

Per a documentació de classes: **@version** <informació sobre la versió>

**@author** <nom autor>

Per a documentació de mètodes: **@param** <nom argument><descripció>

**@return** <descripció>

**@exception** <excepció>

Per generar la documentació: **javadoc nom.java → nom.html**

O bé des de el Netbeans, botó dret damunt del nom del projecte



- Paquets

Les classes (i interfases) estan agrupats en paquets. Un fitxer .java només pot contenir:

- una sentència **package**
- sentències **import**
- definició de class i/o Interface
- comentaris

La resta (variables, constants, mètodes) estarà dins de les definicions de classe o Interface. Cada classe o interface en Java esta dins d'un paquet que es declara al principi del fitxer que de la classe o interface:

**package** nom; //declara tot el que hi ha al fitxer com part del paquet "nom"

Si no es declara un paquet específic a llavors Java treballa amb un per defecte (default) que no te cap nom.

Els paquets es poden ennuiar formant una jerarquia tipus:

*paquet.subpaquet.subpaquet.classe*

Per exemple: *java.lang.System.out* segueix la convenció per anomenar paquets de manera que s'assenyala la variable *out*, de la classe *System*, del paquet *lang*, del paquet *java*. Que a més a més determina un subdirectori del disc: *java/lang/System/out/* a partir del PATH on estigui instal·lat java.

Exemple:

```
package upc.cs.alfons;
import java.util.Date;

// La classe PrintData estarà declarada dins del paquet upc.cs.alfons
class PrintData {
    public static void main (String[] args) {
        Date ara = new Date();
        System.out.println(ara);
        // La classe System esta dins del paquet java.lang
        //(que s'importa per defecte). Podria posar-se:
        //java.lang.System.out.println
    }
}
```

## BASE PRÀCTICA

- **Programes d'exemple.**

1. Editar, compilar i provar que el següent programa visualitza per pantalla el missatge: "Hola Mon, que tal estàs?".

```
// Primer programa: HolaMon.java
/** @author Alfons Valverde
    @version 1.0 */
public class HolaMon {
    /** @param args sense arguments a la línia de ordres (consola d'entrada)
        @return res */
    public static void main(String[] args) {    // El programa comença sempre al main()
        System.out.println("Hola mon ");
        System.out.print("que tal estàs? ");
        // System.out.print("Hola Mon, que tal estàs?");
        // System.out.print("Hola Mon, " + "que tal estàs?");
    }
}
```

2. Editar, compilar y provar que el següent programa calcula el perímetre de la circumferència de radi  $r$ , introduït aquest des del teclat.

```
import java.util.Scanner;

public class AreaCercle {
    // static final double PI = 3.14159265;
    public static void main (String [] args){
        int radi;
        double perímetre;
        Scanner input = new Scanner(System.in);
        System.out.println("Introduiu el radi: ");
        radi = input.nextInt();
        perímetre = 2*Math.PI * radi;    //3.14156
        System.out.println("L'area del cercle es de: " + area);
    }
}
```

3. Comproveu que el següent programa calcula la mitjana de dos números enters.

```
import java.util.Scanner;

public class Mitjana {
    public static void main (String [] args){
        int a, b;
        Scanner input = new Scanner(System.in);
        System.out.println("Introduiu dos números enters: ");
        a = input.nextInt();
        b = input.nextInt();
        System.out.println("La mitjana es: " + (a+b)/2.0 );
    }
}
```

4. Editar, compilar y probar que el següent programa calcula el total a pagar per una compra un cop demanats a l'usuari el preu de l'article i les unitats desitjades.

```
import java.util.Scanner;

public class TotalCompra {
    public static void main (String [] args){
        double preu,total;
        int unitats;
        Scanner input = new Scanner(System.in);
        System.out.println("Unitats desitjades: ");
        unitats = input.nextInt();
        System.out.println("Preu unitari: ");
        preus = input.nextDouble();
        total = unitats * preu;
        System.out.println("Total a pagar: " + total + " euros.");
    }
}
```

5. Editar, compilar y probar que el següent programa que calcula el quocient i el residu de la divisió sencera de dos números..

```
import java.util.Scanner;

public class DivisioInt {
    public static void main (String [] args){
        int a, b;
        Scanner input = new Scanner(System.in);
        System.out.println("Introduiu el divisor: ");
        a = input.nextInt();
        System.out.println("Introduiu el dividendi: ");
        b = input.nextInt();
        System.out.println("Quocient: " + a/b + " , Residu: " << a%b);
    }
}
```

6. Comproveu que el següent programa ens diu el codi ASCII d'un caràcter.

```
import java.util.Scanner;

public class Ascii {
    public static void main (String [] args){
        int num;
        Scanner input = new Scanner(System.in);
        System.out.println("Introduiu un número enter: ");
        num = input.nextInt();
        System.out.println("El codi: " + num + " es correspon al caràcter: " +
                           (char)num );
    }
}
```

- **Programes per completar**

- 1.- Modifiqueu el programa 1 de manera que mostri el missatge “Hola” i “Adeu” en dues línies consecutives.
- 2.- Modifiqueu el programa 2 de manera que es calculi l'àrea d'un cercle:  $\pi \cdot r^2$ .
- 3.- Modifiqueu el programa 3 de manera que donats la velocitat mitjana d'un cotxe i la distancia a recórrer es mostri el temps que trigarà a fer-la.
- 4.- Modifiqueu el programa 4 de manera que en el total a pagar es tingui en compte el 16% d'IVA.
- 5.- Modifiqueu el programa 5 de manera que faci servir la conversió de tipus explícita.
- 6.- Modifiqueu el programa 6 de manera que l'usuari introdueixi un caràcter i digui quin és el seu codi numèric segons la codificació ASCII.

- **Programes per desenvolupar**

- 1.- Implementeu un programa que demani l'edat i el nom de l'usuari i a continuació mostri el missatge: “Hola Pep, tens X anys”.
- 2.- Implementeu un programa que calculi l'àrea, el perímetre i la longitud de la seva diagonal d'un quadrat, donat el seu costat. (Nota: Per fer arrels quadrades feu servir la funció (mètode) `Math.sqrt(num)`)
- 3.- Implementeu un programa que permeti convertir una temperatura donada en graus Celsius a una temperatura en graus Fahrenheit:  $F = 9/5(C+32)$ .
- 4.- Implementeu un programa que faci la conversió de minuts a hores i minuts. L'usuari introduirà un nombre enter de minuts i es visualitzaran el nombre de hores i minuts corresponents.

- **Programes opcionals**

- 1.- Implementeu un programa que donada una quantitat en segons, la transformi a hores, minuts i segons.
- 2.- Implementeu un programa que demani una quantitat en cèntims d'euro i ens faci la descomposició en monedes d'un euro, de 50 cèntims d'euro, de 10 cèntims d'euro i d'un cèntim d'euro.
- 3.- Implementeu un programa que donat un nombre enter de quatre xifres faci les descomposició en unitats de mil, centenes, desenes i unitats.

